

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A) 平3-127236

⑬ Int.Cl.⁵
G 06 F 11/32

識別記号 庁内整理番号
B 8522-5B

⑭ 公開 平成3年(1991)5月30日

審査請求 有 請求項の数 6 (全17頁)

⑮ 発明の名称 トレースデータを図形形式に変換するためのコンピュータが実現する方法

⑯ 特 願 平2-239774

⑰ 出 願 平2(1990)9月10日

優先権主張 ⑱ 1989年10月13日 ⑲ 米国(US) ⑳ 420845

㉑ 発 明 者 チャールズ、アンドルー、ルーカ アメリカ合衆国カリフォルニア州、サラトガ メリツク、ドライブ、20210

㉒ 出 願 人 インターナショナル、ビジネス、マシーンズ、コーポレーション アメリカ合衆国ニューヨーク州、アーモンク(番地なし)

㉓ 復 代 理 人 弁理士 佐藤 一雄 外4名

明 細 書

1. 発明の名称

トレースデータを図形形式に変換するためのコンピュータが実現する方法

2. 特許請求の範囲

1. 仮想プロセッサを同時に実行することから得られるトレースデータを図形形式へ変換するためのコンピュータが実現する方法において、

(a) 与えられた任意の並列タスクに対する並列活動と呼出す引続く事象の連結されたリストへ前記トレースデータを再書式化する過程と、

(b) 前記リストのリンクを選択的に横切り、プロセッサ利用と並列活動構造の階層の任意の実行との、前記横切りと同期した、時間処理表示を行わせる過程と、

を備えるトレースデータを図形形式へ変換するためのコンピュータで実現する方法。

2. 1つまたは複数の仮想プロセッサに対するアプリケーションのタスクの同時実行を示すオペレーティングシステムディスパッチャーにより発生されるトレースデータを図形形式へ変換するためのコンピュータが実現する方法において、

(a) 仮想プロセッサと、タスクIDと、並列活動と、各トレースデータ事象に対する1つまたは複数の並列活動メトリックスとを識別およびクロスリンクするトレースデータ(31、31')から表(33、35、31')を形成する過程と、
(b)(1) プロセッサ利用のためにプロセッサおよび活動クロスリンク(63~93)により指標付けられるものとして、および

(2) 並列タスクの階層を実行するためにタスクおよび活動クロスリンク(99~119)により指標付けられるものとして、

表要素を表示装置に図形的に表示する過程と、を備えるトレースデータを図形形式へ変換するためのコンピュータが実現する方法。

3. 請求項(2)記載の方法において、過程

特開平3-127236(2)

(a) と (b) は、

(a 1) タスク ID と、このタスク ID において分類されている並列活動との用語索引 (35) を取出す過程と、

(b 1) 並列タスクの階層を実行することの表示への表要素のマッピングにおいて用語索引により表をアクセスする過程と、

をそれぞれ含む方法。

4. 請求項 (2) 記載の方法において、表示へ表の要素を図形的にマッピングする過程は、図形データ表示マネジャー (GDDM) を呼出す過程と、選択されたプロセッサとタスク ID および活動により指標づけられて前記 GDDM により表示に色をつける過程とを含む方法。

5. 請求項 (2) 記載の方法において、表示は仮想プロセッサ利用を示す時間処理図を含み、前記マッピング過程は、処理図が、時間範囲すなわち間隔として表されるズーム、前方視点および後方視点にできるたびに、各仮想プロセッサ ID ごとに 1 回前記表を処理することを含み、

(33, 33', 35) を作成し、かつ置き、平均同時実行を計算し、表の項 (34, 34') をクロスリンクさせる過程と、

(c) クロスリンクの選択されたサブセットにより指標づけされたデータ (活動数とポイント) を表から取出し、その取出したデータを視覚的に知覚可能な媒体にマッピングする過程と、

を備えるアプリケーションプログラムを N 個の仮想プロセッサで並列に実行することについての情報を処理および表示するためのコンピュータが実現する方法。

3. 発明の詳細な説明

[産業上の利用分野]

本発明は並列プログラム実行の評価に関するものであり、更に詳しくいえば、トレースデータに記録されている個別機能事象を、並列活動の持続時間と、並列処理要約変数と、評価目的のための同時変数とを示す時間処理線図への変換に関するものである。

前記マッピング過程は、第 1 のスレッドおよび第 2 のスレッドの態様で並列活動を表す過程と、各第 1 のスレッドに対する開始時刻と終了時刻を選択された視点の時間範囲 (T_{min} , T_{max}) と比較する過程と、可視性を高めるためにスレッド活動を図の時間範囲に一致させるために視点を調節させる過程とを含む方法。

6. アプリケーションシステムまたはオペレーティングシステムに埋込まれている機能をディスパッチすることにより発生されたトレース記録 (31, 31') から得られたアプリケーションプログラム (21~23) を N 個の仮想プロセッサ (15~19) で並列に実行することについての情報を処理および表示するためのコンピュータが実現する方法において、

(a) 前記プロセッサにより発生されたトレース記録を走査することにより並列活動を確認および認識する過程と (典型的な事象は FORK, スケジュールされたタスク等を含む)、

(b) 識別された並列活動から時間順序事象の表

[従来の技術]

コンテキスト

本発明は並列活動の評価を処理するものである。この処理は直列に行われる処理よりも複雑である。これは、共通のアプリケーションに属し、独立した仮想プロセッサにより処理されるいくつかのオブジェクトが、処理順序への予測されないデータ依存性と、資源スケジューリング制限を不断に示す傾向があることに由来するものである。

興味のある第 1 の特質は、並列処理されるオブジェクトが決定論的なものであるか、否かということである。次に、同時または順次の再生評価を行えるようにするために、適切な並列処理事象の記録についての関心がある。最後に、本発明の目的のために、並列処理状態の情報を選択することと、それを有用な評価形式へ変換するという問題点がある。

システムおよびコードにおける決定論および非決定論

システムすなわち実行可能なコードのボディは、

特開平3-127236(3)

ムに余分な負荷をかける。

選択されたシステムにおけるログ、トレースおよび

び反復履歴

ログとトレースは、システムの情報と制御情報

の少なくとも一方への変化を表す記録された履歴

の時系列を意味する。ログ記録は、障害時に情報

の一貫性の以前の状態を再び制定するために、ト

ランザクシヨ向き管理システムに関連して長い

間用いられてきた。1985年2月5日付けで登

定されたベーカー(Baker)他の米国特許第4,498,

145号「データベース装置における複数行更新オ

ペレーシヨンのアトミシチを保證する方法

(Method of Assuring Atomicity of Multi-Row

Update Operations in a Database System)」を

参照されたい。

プログラム列の実時間デバッギングと、プログ

ラムおよび計算環境相互作用を示す研究のため

にトレースが用いられてきた。順次実行されるオ

ラジエクトのデバッギングの場合には、目的コー

ド中の点を、そのコードと対をなす原始コードに

それに関連される事象の履歴を反復することによ

り、それを同じ状態および結果へ駆動できる場合

に決定論的である。しかし、資源の利用可能性の

変化により、システムの状態変化の相対的な順序

または絶対的な順序が変化させられる場合には、

履歴を反復してもシステムが同じ状態へ駆動され

ることは必ずしもならない。この場合には、シ

ステムすなわちコードのホイズは非決定論的であ

る、とみなされる。

たとえば、並列計算環境において実行する規則

を基にした推論およびアトリケーションは非決定

論向きのシステムである。前者は、次の実行サイ

クルのためのデータセットにおける変化により運

択された規則セットを、2つまたはそれ以上の結

果へ駆動できるという事実によるものである。ま

た、後者は、計算が非可換性オペレーションとデ

ータ依存を含むことがあるから、アロセッサの間

の利用可能性の変化に敏感である。したがって、

並列処理における非決定論は、単なる順次処理と

は対照的に、デバッギングのような評価メカニズ

同期させることを助けるため、および同期された

点を有意なやり方で表示するためにトレースが用

いられる。命令的な言語作用におけるデバッギン

グにトレースを用いるためには、1988年3月

3日に特許査定されたサイトーほかの米国特許第

4,730,815号「プログラムを試験する図式法(Di-

agrammatic Method of Testing Programs)」を参

照すべきである。この米国特許にはフォートラン

原始コードと、その向けられた図形表現とをス

テップモード、編集モードおよび再試験モードで

相関させる方法が開示されている。

また、1989年4月11日に特許査定された

デュイスバーク(Duisberg)の米国特許第

4,821,220号「プログラムオペレーションをアニ

メ化し、時間を基にした関係で表示する装置

(System for Animating Program Operation and

Displaying Time-Based Relationships)」も参

照すべきである。この米国特許に開示されている

技術は、アトリケーションモデルとして動作する

目的向きプログラムミシヨシステム(OOPS)に

流れとの間に同期が再び行われる。

いることによる1つの規則から別の規則へ制御の

よりセットされる規則の衝突と、1段階指令を用

びポストアトリケーション規則表示状態を用意すること

に、コードのセグ

メントをデバッグするための2段階法を用いる。

れた規則を基にした推論のために、コードのセグ

いる技術は、手続きコードのセグメントに埋込ま

を参照されたい。この米国特許出願に開示されて

ツツ(Two Step Debugger for Expert Systems)」

/245,475、「エキスパートシステム用の2段階デバ

ーネル(Darnell)他による未決の米国特許出願07

最後に、1988年9月16日に出願されたタ

も1つの可視表示がある。

の態様でトレースを用いる。状態変化の少なくと

された事象(時間的に順序づけられたメッセージ)

ライズするために、順序づける明らかな時間記録

において相互作用オラジエクトの間で状態変化をド

478

特開平3-127236 (4)

ユニプロセッサにおけるプロセス、スレッド、同期および同時実行

アメリカ合衆国カリフォルニア州サンタ・クルス (Santa Cruz) 所在のカリフォルニア州立大学のコンピュータおよび情報科学教育局 (Board Of Studies in Computer and Information Science) により1988年2月4日に発行されたマクドウェル (McDowell) およびヘルムボールド (Helmbold) 著「並列プログラム用のデバッグツール (Debugging Tools for Concurrent Programs)」の27～32ページに指摘されているように、順次処理は1つの順次出力装置により解析されるのに役立つ。これに関して、順次プログラムは実行の1つのスレッドだけを表す。これは順次テキストとして正確に表示できる。これとは対照的に、並列処理は多数のスレッドを含む。この場合にはデータを論理的および物理典型的に分布させることができる。

アディソン・ウェズレイ・パブリッシング・カンパニー (Addison-Wesley Publishing Company)

がpに達し、Qがqに達する順序、すなわち、処理の間の制御情報の交換、を拘束するために同期化が用いられる。

この本によれば、並行計算を事象で記述できる。ここに、「事象」というのは中断されない活動である。すなわち、事象というのは手続き言語が原子として取扱う任意の構造である。したがって、処理のスレッドは一連の事象に対応する。

相対的な順序付けおよび同時実行

処理AとBがステップ「a b c d e」と「w x y z」をそれぞれ含み、単一プロセッサで独立に実行する場合には、相対的な順序が保たれている限り、インターリーブされた任意の順序を許容できる、すなわち、「a b w x c y j d e」と同様に「a w b x c y d j e」を許容できる。データまたは順序によるように依存性、すなわち、サブルーチニング、コールルーチニングまたはネスティング、が存在する場合には、インターリーブされた順序をほぼ制約できることが明らかである。

により発行され、ベル研究所 (Bell Laboratories) が1989年に著作権を取得した、ラビ・セジ (Ravi Sethi) 著「プログラミング言語—概念および構造 (Programming Languages—Concepts and Constructs)」の343～379ページ「並行プログラミング入門 (An Introduction Into Concurrent Programming)」には、「処理 (Process)」がそれ自身の制御スレッドで順次計算に対応する、と主張されている。順次計算の「スレッド」は資源コードの片割れを横切ることにより測定される一連のプログラム点である。この本には、処理の間の相互作用が、明示メッセージにより、または共用されている変数の値によるデータの交換を含むことが更に指摘されている。ある変数が処理にとって見るることができるものであるならば、その変数は処理の間で共用される。

並行処理に関しては、同期化によって1つの処理のスレッドが別の処理のスレッドに関係づけられる。pを処理Pのスレッド中の1つの点とし、qを処理Qのスレッド中の1つの点とすると、P

並列処理および言語拡張

「並列処理」という用語は、作業の単位が仮想プロセッサの使用可能性の関数としてスケジュール/指命される複数プロセッサ環境におけるアプリケーション実行を意味する。アプリケーションは、並行して実行可能な作業の単位の呼出し、実行、同期化および終了を定める構造を含むために拡張されたV S フォートランのような高レベル言語の編集された態様で表すことができる。この拡張には次のものが含まれる。

並列プログラム

並列ループ	— 並列反復
並列部	— 並列ステートメント
並列呼出し	— 並列サブルーチン
オリジネイト、...	
スケジュール、]
待機、]
終了、]

並列処理、トレース分析、デバッグ

目的 (デバッグ) の正しさ、または計算資

特開平3-127236(5)

源使用の効率(同調)を変更することは、必然的にトレース情報に依存する。従来技術は、いずれか、または両方の活動を行うことができる全てを含む枠組みを設けようという試みで満ちている。

アイビーエム・リサーチ・レポート(IBM Research Report)RC 13662、1988年4月、所載のジャニス・ストーン(Janice Stone)の「並列処理の図形表現(Graphical Representation of Concurrent Processes)」と題する論文には、正確であることを確認するため(デバッグ)に、事象のアニメ化された再生を行うために同時マップとトレースを使用することが開示されている。この論文の同時マップは、処理の履歴を時間格子上に表示し、処理間依存性により定められた依存性ブロックをそれから取出すことにより形成された1組の相関させられた事象の流れである(上記論文の4ページ参照)。この論文には、同時マップは全ての可能な事象順序づけの必要かつ十分な記述であることが断言されている(5ページ)。また、マップは、応答はもちろん、図形活動表示

を制御するためのデータ構造として機能する。

1988年3月25日にナサ・エームス・リサーチ・センター(NASA Ames Research Center)へ提出されたローレンス・リバーモア・ラボラトリイ・レポート(Lavrense Livermore Laboratory)UCID 21348、所載のマーク・ケー・シーガー(Mark K. Seager)他の「図形多重処理解析ツール(GMAT)(Graphical Multiprocessing Analysis Tool)」と題する論文には、大域状態遷移図の形式での並列処理を示して、時間ラインおよびアイコン(icon)を強調する。

[発明が解決しようとする課題]

したがって、本発明の目的は、1つまたは複数の仮想プロセッサに対して並行に実行する計算に関して、資源の利用および並列を評価するコンピュータが実現する方法を提供することである。

本発明の別の目的は、並列処理概要変数と、並列活動持続時間と、同時処理変数を取り出すために、並列実行中に起き、仮想時間ではなくて実時間で記録される機能的な事象を分析するコンピュ

ータで実現する方法を提供することである。

本発明の更に別の目的は、(1)負荷を釣合わせる等のことにより並列アプリケーションの性能を向上させ、(2)並列ループが並列または直列に実行されるかのような並列実行経路を検査し、(3)実行時間データおよび資源依存を識別することによるというようなデバッグを行う、ことの援助として資源の利用および並列を図形的に示すコンピュータが実現する方法を提供することである。

[課題を解決するための手段]

並列活動を開始および終了させる事象がリンクされるように、トレースデータの選択された部分が再書式化されたとなると、時間処理線図を得て表示するために、再書式化されたデータの検討を使用できることが予期しないことに観察された。そのような再書式化およびリンクは、計算および仮想プロセッサの状態を識別する他の要因が、並列活動を呼出す事象に高く相関させられる、という事実を利用する。他の要因にはプロセッサ、タ

スクID、時間要因が含まれる。また、(1)活動状態、アイドル/待機状態、または使用不可能状態による仮想プロセッサ利用と、(2)並列タスクの階層を実行する要素の機能的な分解と、を示すための時間処理線図を作成するために、再書式化されたトレースデータの検討または分解を使用できることも観察された。

より簡潔に言えば、前記目的は、仮想プロセッサを並行処理することから得たトレースデータを図形の形に変換するための、コンピュータが実現する方法により達成される。この方法は、(a)仮想プロセッサと、タスクIDと、並列活動と、1つまたは複数の並列活動マトリックスを識別およびクロスリンクする前記プロセッサにより発生されたトレースデータから表を形成する過程と、(b)表の要素を、(1)プロセッサ利用を示すためにプロセッサおよびクロスリンクにより指標付けられ、(2)および並列活動の階層の実行を示すために活動クロスリンクにより指標付けられる、ものとして表示器に図形的にマップする過程

特開平3-127236(6)

と、を備える。

その表は発生時刻と、仮想プロセッサと、タスクIDと、並列活動を開始する少なくとも1つの事象と、現在の事象により開始された並列活動のための終了事象に対するポイントとを相関させる。プロセッサ利用表示器は、仮想プロセッサIDについての第1の分類と、表の事象トレースについての第2の分類とにより駆動される。同様に、実行並列タスク階層表示は、タスクIDについての第1の分類と、リンクされた活動についての第2の分類とにより駆動される。また、タスクIDにより表に反転されたインデックスを用いることにより、並列活動表示を促進できる。

ストーンの論文と本発明の対比

ストーンの論文の8～10ページに記載されているアニメーションは、正確さを向上するため(デバッグ目的)に表示を動的にすることに向けている。したがって、ストーンの論文において、仮想プロセッサの並行実行の間での負荷の釣合いを表示するものとする、それは一連の疑似

び終了させられることが予測される。しかし、ひとたび発生されると、タスクは繰返し呼出される。呼出しは、サブタスクが終ることを待つためにタスクを引き続いて呼出すことを求める態様で行うことができる。あるいは、呼出しているタスクによりはっきりと待つことなしに、サブタスクが実行を終ることを可能にする態様で呼出しを行うことができる。

フォートランプログラムは、従属するタスクの創造、取扱いおよび最後に削除することを典型的に含む。各タスクは独立している。呼出すタスクと、呼出されるタスクの間で共用されるデータを除き、および2つのタスクにより共同して取扱われるファイルを除き、1つのタスクの実行は別のタスクの実行とは理想的に独立しており、かつ別のタスクの実行に影響を及ぼすことができない。

これ以上の詳細が下記のIBM出版物に記載されている。VSフォートラン・バージョン2プログラマの案内リリース5(IBM Publication SC26-4222-5)およびVSフォートラン・バージョン・

ランダムな急激な遷移として現われるであろう。これとは対照的に、本発明の方法は同調を助けることにある。それは、時間的に個別の点にわたる、平均的なプロセッサ活動状態、アイドルまたは待機状態の分布のような測定された変数または重みづけられた変数を示す。それらの点は、並列活動のスレッドの指定されたフォークおよび連結部に生ずる。すなわち、この方法は並列処理の終点に起こる活動へ向けられる。その表示は、仮想プロセッサIDまたはタスクIDについて第1に分類され、第2に事象について分類される再書式化されたトレース内に埋込まれているクロスリンクの検討に同期させられる。

[実施例]

フォートランの面および言語拡張を用いる並列処理

並列実行のために拡張されたフォートラン言語は、主プログラムすなわちタスクがサブタスクの発生、呼出しおよび終了をできるようにする構造を典型的に含む。タスクは比較的にまれに発生およ

2言語およびライブラリ参考マニュアル・リリース5(IBM Publication SC26-4221-5)

(VS FORTRAN Version 2 Programmer's Guide Release 5(IBM Publication SC26-4222-5)、VS FORTRAN Version 2 Language Reference Manual Release 5(IBM Publication SC26-2221-5))。

また、フォートラン言語系へ拡張する説明が並列構造と、タスクの間の通信を強調するためのものである、1988年5月20日に出願された「フォートランに似た言語系から編集された共通ブロックを同時に順次アクセスするタスクの間の衝突を減少する方法(Method for Reducing Conflict Accessing Common Blocks in Sequences Compiled From a FORTRAN-like Language System)」と題する、カープ(Karp)他による未決の米国特許出願第07/197,060号も参照されたい。

あるフォートラン拡張

第1図に示されているタスク階層と、第3図に示されている並列スレッド階層とをサポートするため、および、たとえば、前記IBMフォートラ

特開平3-127236(7)

ン・バージョン2リリース5言語系に見出だされるように、選択的な並列オペレーションを確実に行うために、フォートラン言語の拡張が用いられる。構造は下記のように定義される。

並列プログラムは、1つまたは複数の実行可能なオブジェクト（プログラム）、自動並列実行のために編集されるオブジェクト、またはロックまたは事象サービスルーチンを用いるプログラムである。

並列ループ（DOに等しい）は、指定された条件が存在する限り実行される1組のステートメントを含む。ループの各繰返しはループの他の繰返しと同時に実行できる。有効なオペレーションは計算の独立性、または修正され、共用されるデータ区域に対する制御されたアクセスを要求する。対照的に、直列処理されるループに対する繰返しの順序は一般に制限され、または予め定められる。

並列ループは並列DOステートメントにより明確に形成されるか、自動的に並列にされるDOループにより発生される。ループの各繰返しは仮想

プロセッサにより実行される並列スレッドとすることができる。また、繰返し群は、仮想プロセッサで実行される1つのスレッドに組合わせることができる。並列ループ実行に同時に関与する仮想プロセッサの数は、実行すべき他の並列作業の量に依存する。たとえば、10000回の繰返しの並列群を、それぞれ2500回の繰返しよりなる4つの群に区分できる。それらの繰返しは2つの仮想プロセッサで実行できる。これに関して、1つの仮想プロセッサは3つの群を実行し、他の仮想プロセッサが1つの群を実行できるようにできる。

並列部分は順次実行されるステートメントの群である。各部分は並列部分中の他の部分と同時に実行できる。

並列タスクは、データの記憶を含めた実行のために並列スレッドが完全に指定されるような、摘要データ形式である。ルートタスクがプログラムの実行の開始点を指定する。付加並列タスクが初めのステートメントにより形成され、スケジュールステートメント、待ちステートメントおよび終

りステートメントにより管理される。

ここで、複数レベルの並列タスク1～13が示されている第1図を参照する。複数レベルの並列タスクが作成されると、階層的な所有権が制定される。発生されたタスク7、9、11は片割れの並列タスク3、5によって「所有される」。並列タスク3、5においてはタスクが発生したステートメントが指定されていた。また、タスク9はタスク13の親である。最後に、ルートタスクは並列タスク構造の最高レベルである。

並列スレッドは並行実行のために適格な作業の単位である。並列スレッドはサブルーチン、ループ繰返し、またはコードの部分である。それらのおのおのに対しては、スレッドはそれらの間で呼出された機能と、それらの間に呼出され、呼び出しステートメントを有するサブルーチンとを含む。複数のスレッドを同時に実行することにより並列処理が行われる。並列言語ステートメントは、並行実行のために1つまたは複数の付加並列スレッドをディスパッチできる。そのようなディスパッ

チされた、すなわち子供の並列スレッドは親スレッドの一部とは考えられず、別々に実行できるものと考えられる。重要なことに、並列プログラムが実行される時は、それはその第1の並列スレッドで始まる。

並列スレッドが並列のDOステートメント、部分ステートメントまたは呼出しステートメントにより限られている時に、それらの並列スレッドがディスパッチされるのと同じタスクで並列スレッドは実行する。また、スケジュール構造が用いられる時に並列スレッドがディスパッチされたものから、並列スレッドは種々の並列タスクで実行できる。

次に、固有の非決定論処理すなわち並列処理が示されている第3図を参照する。これは2つのソース、すなわち、タスクの間の同時処理／待機関係、および仮想プロセッサの使用可能性、から得る。この図においては、相関させられる3つの垂直ラインとして表される。時間は上から下へ増大する。左上には並列DOが述べられている。これ

特開平3-127236(8)

は何百というスレッドの形で表すことができる。親タスクから発生された並列D Oの場合には、並列D Oのスレッドが実行されるまで、名目的に期は延期される。

再び第3図を参照して、次の2行は発生するタスクと、子タスクまたは独立に発生されたタスクとのその発生するタスクの時間の重なり合い関係すなわち同期処理関係とを示す。これに関して、発生するタスクのスレッドは延期でき、または重なり合わされた関係で継続できる。フォートランにおいては、発生するタスクが子タスク、およびその発生するタスクのルートタスクに対しても、関連してそれ自身を延期する点を制御するために待ち構造が用いられる。中間行は、全てを待つ構造が呼出されたタスクの呼出しの初めに実行される場合を示し、下の行では、待ちはルートタスクを階層的に従属するタスクの実行へ十分に延長させる。

仮想プロセッサおよび実行のスレッド

次に、仮想プロセッサ15~15と、並列タス

クサービスおよび事象サービスを用いて、とくにデータ依存性が存在する時に、ある程度の順位を定めることができることを理解すべきである。MVSのようなバッチ向きオペレーティングシステムにおいては、仮想プロセッサはMVSタスクに対応する。同様に、VMに類似するオペレーティングシステムにおいては、仮想プロセッサは仮想CPUに対応する。仮想プロセッサを実プロセッサへ結びつけて、スケジュールするのは、編集されている実行時間コード(アプリケーション)ではなくてオペレーティングシステムであることに注目されたい。

並列プログラムが1つの仮想プロセッサで実行する時は、並列スレッドは同時には実行しない。むしろ、並列スレッドは疑似直列で実行する。スレッドの一部を実行でき、別の並列スレッドを実行できるように、同期化要求のために実行を延期できる。第2のスレッドが終わった後で元の並列スレッドを回復できる。そのような処理においては、スレッド内の事象またはステップの相対的な

ク21~23と、付随するスレッドと、ローカル環境25~29の間の動作関係が示されている第2図を参照する。各仮想プロセッサは、それにおいて第1の平行スレッドが実行される論理プロセッサである。実行が進むにつれて、並列言語系によって、他の並列スレッドが第2図のようにしてディスパッチされる。フォートランは編集される言語系として、仮想プロセッサの数と管理を含む実行時間(ローカル環境)環境を指定する。しかし、実プロセッサへの仮想プロセッサの結合はMVSのようなオペレーティングシステムにより制御され、編集される並列フォートランをベースとするアプリケーションではない。

複数の仮想プロセッサが指定されると、ディスパッチされた並列スレッドを同時に実行できる。全ての仮想プロセッサが活動状態にあるとすると、仮想プロセッサを使用できるようになるまで並列スレッドは実行を待つ。並列スレッドの実行順序は、並列プログラムの1つの実行と別の実行の間で変えることができる。明らかな言語構造とロッ

順序を保持する必要があることを思い出されたい。実行を開始した並列スレッドとプロセッサの間に1対1の対応を維持するために十分でない仮想プロセッサが存在する時に起るこの処理は「再ディスパッチング」と呼ばれる。

表の作成

次に、本発明の基の典型的なトレース構造と処理構造が示されている第4図を参照する。トレース31は、並列処理事象に関連する属性の時間スタンプが分類したログである。事象の形式はフォークと結合を含む。古い文献(アディソン・ウェズレイ出版社(Addison-Wesley Publishing Company)により1978年に出版された、ホルト(Holt)他著「オペレーティング・システム・アプリケーションズによる構造化された並行プログラミング(Structured Concurrent Programming With Operating Systems Applications)」18、31ページ)には、フォークは並行処理タスクへの分岐と示され、結合は終点であった。記録されている属性は仮想プロセッサIDと、事象の種類

特開平3-127236(9)

と、タスクID等を含む。さて、本発明の方法の1つのステップはトレースを再書式化し、再処理することである。その結果としての構造33、35が並列活動および並列処理法とそれぞれ名づけられる。活動表33中のトレース事象に対する再書式化された各エントリイは独特の活動番号と、現在の事象で開始させられた並列活動に対する終了事象に対するポインタ34とを含む。

並列処理表35は活動表33から得られる。それは、活動表への反転された指標の活動表への混合と、選択された同時処理変数の計算のための記憶装置を構成する。表35の重要な用途は、並列タスクの階層を実行することの表示の発生を促進することに関することである。後者は、処理の開始時に活動表33中の第1のタスクに対するポインタを含み、次にポインタに追従することにより行われる。

次に、実際のトレース31'と、活動表33'と、組立てられた活動表を実行するための制御の流れ図とが示されている第5図と第6図を参照す

る。目的は、トレース31'を再書式化し、活動表33'に類似するフォームで現させるために適切な制御番号とポインタをそれに加えることである。

第5図に示すように、トレース31'は記録7~13を有する。そのうちの記録7~9は並列D0に関連し、記録10、11はスレッドの開始に属し、記録12、13はスレッドの終りに関連する。それに関連して、活動表33'はトレース記録と活動表エントリイを相関させるために注釈されている。たとえば、並列D0のトレース記録は活動表エントリイ45と46へ変換される。

第6図を参照して、表作成の開始は、求められている並列処理の開始を探すためにトレース記録を走査することを服務。これは、開始並列活動に関連する機能事象、すなわち、FORKまたはINIT(第5図においてはPDFORKおよびセシINITと名づけられている)の認識により行われる(ステップ41)。次に、並列処理のために表エントリイ作成を開始する。これはアプ

ケーションの名称、ステートメント番号、並列活動の種類、ポインタ活動タスク表、処理の開始時刻、パラメータおよび初期化データを省く事を含む。

トレースからの次の機能事象の検索(ステップ45)に応答して、この方法は各並列活動のための適切な機能事象終了トレース記録を認識する(ステップ47)。次に、終了記録からのタスクIDを用いて、そのタスクに対してスタートさせられた最後の並列活動が探され、かつその活動状態がセットされた「完成された」を開く。

この方法は、終了時間と、ステートメント数と、ケースラベルと、終了理由とを省くために、並列活動表中の終了エントリイの作成も行う。終了エントリイに対するポインタが、各並列活動のための開始されたエントリイに加えられる(ステップ51)。終了したタスクが「より多くの」状態を有するものとする、この方法は処理に対するより多くの機能的な事象を有する(ステップ53、54、45)。このタスクは同期化ポストにより

終了させられ、ラベル識別子が、より多くの事象の走査の前に活動終了エントリイに格納される(ステップ56、57)。入れ子にされた「ポスト」のために、親タスク名を用いる新しい並列活動が発生され、スタート時刻はポスト時刻であると仮定される(ステップ55)。次の零レベルにおける終了ポストに対して、並列処理は終了したと考えられ、並列活動と並列表が適切な時間と終了データにより更新される(ステップ57、61、65)。

時間プロセス表示が、縦座標アクセスラベルとして各活動の開始ステートメントを、および横ラベルとして並列処理内の時間を用いて時間プロセス表示が発生される。終了ステートメント番号と同期化ラベルが、表示される各時間処理の終りに加えられる。仮想プロセッサ利用の時間プロセス表示が、縦座標アクセスとして仮想プロセッサ番号を、横座標として1つまたは複数の並列プロセスに対する時間ジャイレーションを用いて発生させる。

特開平3-127236(10)

次に、第4図に示されている並列活動表の要約が示されている第10図を参照する。この要約は、各表における連鎖と、主スレッド活動と従属するスレッド活動の間の連鎖とを示す。第10図では、ルート並列タスク主スレッドが従属スレッドを作成するたびに活動表に新しいエントリがある。

第10図において、従属スレッドの作成により、1つの仮想プロセッサから複数の仮想プロセッサへ切替えることによって、並列活動すなわち同時活動が開始される。各主スレッド処理は全ての従属スレッドの終了に対するポイントを有し、それにより制御を1つの仮想プロセッサへ戻すことに注目すべきである。また、各処理エントリは以前の処理にポイントを並列活動中に有する。これは後方スクロールのために使用できる。更に、主スレッド処理のために用いられる各仮想プロセッサは、プロセッサが実行する第1の従属スレッドと最後の従属スレッドのための活動表エントリを有する。最後に、各第1のスレッドエントリは、与えられたプロセッサのための最後のスレ

ドエントリに対するポイントを有する。

最後のスレッドエントリに対するポイントは、1つのプロセッサにより実行される従属スレッドの群のための終了ポイントとして動作する。挿入句的には、各プロセッサにより実行される第1のスレッドのための記録された時刻は、そのスレッドの実行の開始時刻としても作用する。同様に、プロセッサにより実行される最後のスレッドのための記録された時刻は、そのスレッドの実行の終了時刻、またはそのプロセッサの終了時刻である。

表から表示へのマッピング

次に、活動表31'の選択された部分と、表の要素を帰納的に分解し、表示上に図形的にマップするために用いられる制御の流れがそれぞれ示されている第5図と第7図を参照する。第7図におけるステップ67~93は仮想プロセッサ利用の時間プロセス表示の発生に関連するものであり、第7図のステップ91~119は並列タスクの階層を実行することの時間プロセス表示に関するものであることに注目されたい。

次に、時間プロセス線図の独特の種類が示されている第8図と第9図を参照する。第8図は利用の時間ラインを示す。第8図は、プロセッサ活動状態を参照してカラーコード化された片割れの棒チャートも示す。利用に関しては、プロセッサの同時実行変数が、タスク実行のために任意の時刻に用いられる仮想プロセッサの数として定義される。プロセッサの同時実行を時間の関数として積分し、プロセス時間により除してプロセッサの実効数を得ることにより、平均同時実行変数が計算される。重要なことに、この表示により、仮想プロセッサの間で作業の切換えが何回起きたかを再び調べるようにされる。この表示により、並列タスクがプロセッサの間にどのようにして割当てられるか、それらの並列タスクの実行順序等を確かめることができる。

仮想プロセッサ利用線図の発生

次に第8図と第10図を参照する。それは、仮想プロセッサ利用を示す時間プロセス線図が、並列活動表を各仮想プロセッサIDに対して1回処

理することにより作成されるケースである。各主スレッド処理に対する開始時刻と終了時刻が選択された視点の時間範囲(T_{min} 、 T_{max})と比較されて、1つまたは複数の従属スレッド活動を線図時間範囲内で見ることができると決定する。それから、選択されたスレッドに対する開始および終了時刻スレッド実行が線図の時間範囲と比較されて、活動のその部分または全てが線図の範囲内で見ることができるかどうかを決定する。すなわち、

(活動開始時刻 $\leq T_{max}$) および

(活動終了時刻 $> T_{min}$)

活動開始時刻 $\geq T_{min}$ で、活動終了時刻 $\leq T_{max}$ であるとする、図形の表面区域がその視点に対して発生される。開始または終了が線図の時間範囲をこえる活動に対しては、活動の可視部分を表示するために標準的なクリッピング法が用いられる。

T_{min} と T_{max} の線図範囲変数は、表示の適格性についての対応する処理開始番号と処理終

特開平3-127236(11)

了番号を有する。 T_{max} に対応する終了処理に対して可視スレッド活動が図形化されると、本発明の方法は次のプロセッサ利用を図形化するために戻る。

第1のプロセッサ利用線図は、時間範囲が零時刻で始まり、アプリケーション終了時刻で終る、すなわち、 $T_{min}=0$ および T_{max} =アプリケーション終了時刻、線図とすることができる。

与えられた時間範囲のプロセッサ利用線図から、相互作用装置に表示される本発明の方法により、新しい時間範囲(N_{min} 、 N_{max})を有するプロセッサ利用線図を得るために前方または後方へズームまたはスクロールすることをオペレータが選択することが許される。これに関して、3つの表示を最も有利に利用できる。それらはズーム、前方視点および後方視点を含む。

「ズーム視点」は現在の時刻の一部を拡大し、既存の線図時間範囲からユーザーにより選択されるズーム視点時間境界(N_{min} 、 N_{max})において活動データをクリップする。

後のプロセッサ時刻と新しい従属スレッドの次の主スレッド発生間の時間中は、ルート並列タスク主スレッド実行が1つの仮想プロセッサにおいて活動状態にある。

求められているチャートの時間範囲の間の各活動時刻に活動状態にあるプロセッサの総数をプロットすることにより、仮想プロセッサ同時実行図が容易に作成される。用いられる数は、活動表への新しい各エントリにおける各活動表に入れられる活動状態にあるプロセッサ数から計算される。
並列活動図の発生

次に第9図と第10図を参照して、時間プロセス表示を発生するために、活動表から個々の主スレッド(処理番号)を選択することにより並列活動図が作成されることが明らかである。プロセス番号の開始時刻と終了時刻は図の時間範囲に対する T_{min} および T_{max} になる。各仮想プロセッサにおける従属スレッド実行の開始時刻と終了時刻が図形的に発生される。発生順序は活動表で見られる階層番号(H#)を基にする、すなわち、

$$T_{min} \leq N_{min} < T_{max}$$

$$N_{min} < N_{max} \leq T_{max}$$

「前方図」は、新しい線図時間範囲

($N_{max} = N_{min} + T_{max} - T_{min}$)内に並列活動を含む時間範囲に対して、前方へ動く現在の視点におけるある点において新しい視点を開始する。($T_{min} \leq N_{min} < T_{max}$)。

「後方視点」は、新しい線図の時間範囲

($N_{min} = N_{max} - T_{max} + T_{min}$)内に並列活動を含む時間範囲に対して、後方へ動く現在の視点におけるある点において新しい視点をスタートさせる。

$$(T_{min} \leq N_{min} < T_{max})。$$

新しい各プロセッサ利用線図に対して、変数 T_{min} と T_{max} が N_{min} 値と N_{max} 値でそれぞれセットされ、表示の適格性についての対応する新しいプロセス開始番号と終了番号が決定される。

また、各プロセス番号の終了時刻から次のプロセス開始時刻まで、すなわち、従属スレッドの最

同じ番号を有する活動が同じ垂直軸値上にプロットされ、従属スレッドにおける仮想プロセッサの活動の開始と終了はこれの一例である。開始データは活動プロットの左側に列書式で置かれ、終了データは活動の終りに注として置かれる。可変ロックのような一時的なバリエーションが別の例である。挿入句的には、主スレッドの並列階層の時間範囲のズームリングが、仮想プロセッサ利用線図におけるズームリングに類似するやり方で機能する。

再び第9図を再び参照する。この図には並列フォームの階層的な実行によって並列タスク活動が示されている。どのアプリケーションも、入れ子にされている並列実行と複数レベルの並列実行に対して他の並列タスクを実行できることを示すタスク管理構造を用いることにより、複数レベルの並列実行が指定される。実行時刻に実行する並列スレッドの間に定められる階層が示されている。第9図に示されている並列タスク活動の時間処理図により、実行の計画が組まれた時、仮想プロセッサで実行された時、延期させられた時、待つて

特開平3-127236(12)

いた時、およびその実行を終った時、に各並列スレッドを確かめることを許される。

有利なことに、表が選択的な表要素が表示上へ横切られるにつれて、活動表33または33'中の1つまたは複数のエントリに相関させられて表示に「色をつける」ために図形データ表示マネジャー(GDDM)を呼出すことができる。

GDDMは表示端子とプロットに対する物理的インターフェイスを構成し、発明において定められるチャートを示すことを要求される。また、この相互作用は、オペレータ要求、たとえば、ズーム視点に対する仮想プロセッサ利用線図の新規、開始時刻、および終了時刻のオペレータ選択、のキーボード/キーストローク操作により呼出された割込みのGDDM取扱いを利用する。

任意の物理的端末装置すなわちプロット上にチャートを描くことを容易にするために、世界座標系が用いられる。

図形パターンをセット

図形の色をセット

4. 図面の簡単な説明

第1～3図は並列タスクの階層的な編成と、仮想プロセッサの並列プロセッサ編成およびスレッドに対する関係と、拡張されたフォートラン言語で構成された並列スレッド階層とをそれぞれ示し、第4図はプロット典型的なトレースデータと、本発明に従ってそれから取出した処理構造(並列活動表)を示し、第5図は、終了ポイントエントリがトレースの解析中に取出される、並列活動の実際のトレースと付属の並列活動表エントリのスナップショット、第6図は第5図に従って作成された表の制御の流れを示し、第7図はプロセッサの利用および並列分解度すなわち階層に関連する時間の処理グラフへの表の要素の本発明に従ってマッピングするための制御の流れ図を示し、第8図および第9図は時間的なプロセッサ利用の時間処理表示と、時間的な並列活動タスク管理とをそれぞれ示し、第10図は第4図に示されている種類の並列活動表の付加要約図である。

出願人代理人 佐藤 一 雄

優

区域をスタート

長方形の左下をスタート

長方形の左上へ動く

長方形の右上へ動く

長方形の右下へ動く

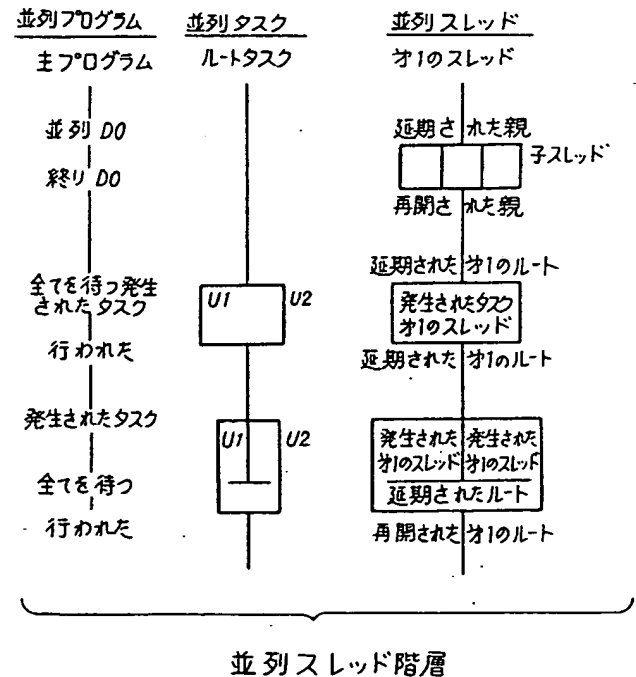
長方形を左下で閉じる

区域を終る

というような、GDDMライブラリ呼び出しを用いる長方形の図形区域の発生のために、

1984年に出版されたアイビーエム・パブリケーション(IBM Publication) SC33-0148、「GDDMアプリケーション・プログラミング・ガイド(GDDM Application Programming Guide)」を参照されたい。

以上、好適な実施例について本発明をとくに説明したが、本発明の要旨および範囲を逸脱することなしに態様および細部をいくつか変更できることを当業者は理解されるであろう。



並列スレッド階層

FIG. 3

31-		時刻 スタッフ	PROJ. NO.	タスク 名	事象の 種類	ソース ISN	事象 状態	事象データ(親の名、 子の名等)
機能事象	#1							
	#2							

事象の種類 =	PT PC PD	PARM FORK INIT EXEC CALL POST WAIT WONE TERM STOP	-S -D	事象の状態 =	FILE TASK CASE WAIT MORE POST CHOP DONE	事象データ =	待ち ラベル待ち ラベルポスト親 親タスク名 子タスク名 チャックサイズ, ITERS 入れ子レベル, PROCS
---------	----------------	--	----------	---------	--	---------	---

34

[illegible]

処理開始時における並列活動の種類
処理の開始時におけるルーチンとステートメント
処理開始時刻、処理の間の時間
処理の開始時におけるサ1のタスクへのポイント
平均周回実行、P/Oセッサ利用
進行中のタスク数
処理経過時間

FIG. 4

特開平3-127236 (14)

記録番号	時刻 スラフ	VIRT PROC	並列 スレッド	事象名	ル-テン	事象特有の情報 ISN	事象記述
7	07365289	6	S0000W0000	PDFORK	FROM	MATMUL.0107	PARM 00 06 00010001 000180
8	07365290	6	S0000.W0000	PDFORK	FROM	MATMUL.0107	TASK S0000.X0001
9	07365292	4	S0000.W0000	PDFORK	FROM	MATMUL.0107	TASK S0000.X0002
10	07365293	4	S0000.X0002	PDINIT	FROM	MATMUL.0107	
11	07365293	6	S0000.X0001	PDINIT	FROM	MATMUL.0107	
12	08259224	4	S0000.X0002	PDTERM	FROM	MATMUL.0113	MORE W0000
13	08265841	6	S0000.X0001	PDTERM	FROM	MATMUL.0113	POST W0000

主プログラムからの並列 D0の実行をトレースする記録

FIG. 5a

並列 D0 構造のために活動表エントリ45と46が作成される
(トレース記録 7, 8, 9)

活動番号	APPL 時間 (μ秒)	VP ID	TOTAL VPS	スレッド活動 ID	COMPL PTR	H#	親 PTR T#	プログラム 名	ISN	PU# PAT	ラベル/ データ
45	07365289	6	1	PD FORK W0000	46	1	0	MATMUL	107	D1	1
46	07365289	6	0	ID PARM W0000							180

スレッド開始のために活動表エントリ47と48が作成される
(トレース記録 10 と 11)

活動番号	APPL 時間 (μ秒)	VP ID	TOTAL VPS	スレッド活動 ID	COMPL PTR	H#	親 PTR T#	プログラム 名	ISN	PU# PAT	ラベル/ データ
47	07365293	4	1	PD INIT X0002	?	2	45 0	MATMUL	107	D1	
48	07365293	6	2	PD INIT X0001	?	3	45 0	MATMUL	107	D1	

FIG. 5b

スレッド終了のために活動表エントリ49と50が作成される
(トレース記録 12 と 13)

活動番号	APPL 時間 (μ秒)	VP ID	TOTAL VPS	スレッド活動 ID	COMPL PTR	H#	親 PTR T#	プログラム 名	ISN	PU# PAT	ラベル/ データ
49	08259224	4	1	ID TERM X0002		2	45 1	MATMUL	113	D1	
50	08265841	6	0	ID POST X0001		3		MATMUL	113		

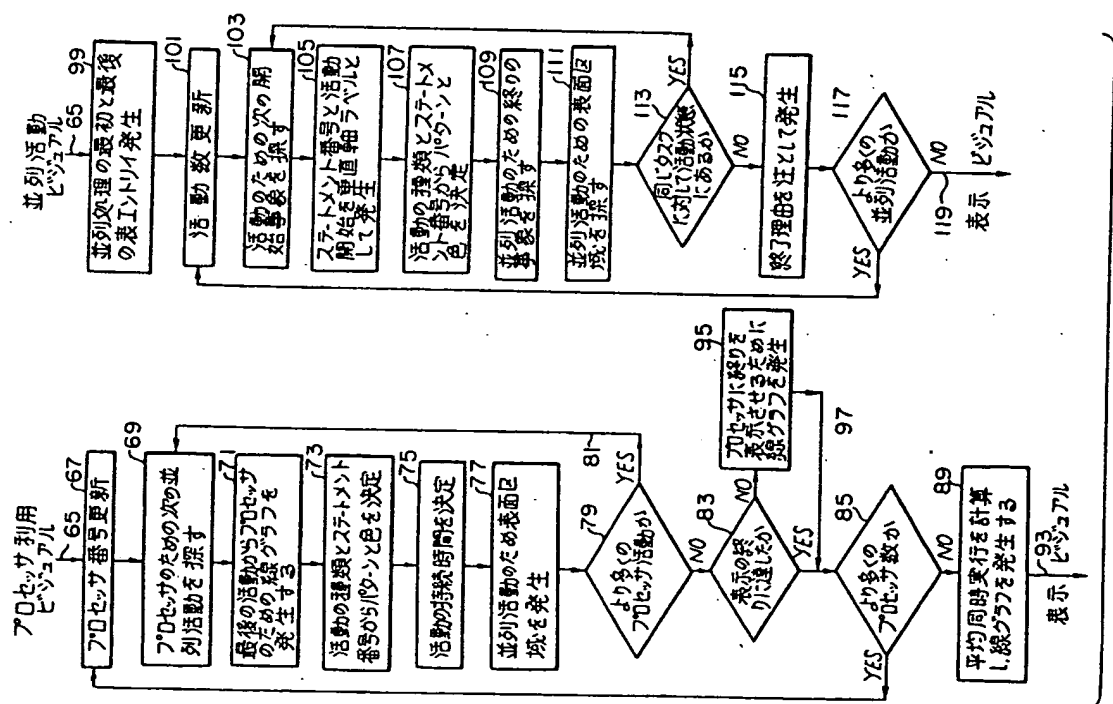
終了スレッド2と1のPTRを終了させるために活動表エントリ47と48が更新される

活動番号	APPL 時間 (μ秒)	VP ID	TOTAL VPS	スレッド活動 ID	COMPL PTR	H#	親 PTR T#	プログラム 名	ISN	PU# PAT	ラベル/ データ
47	07365293	4	1	PD INIT X0002	49		45 1	MATMUL	107	D1	
48	07365293	6	2	PD INIT X0001	50		45 1	MATMUL	107	D1	

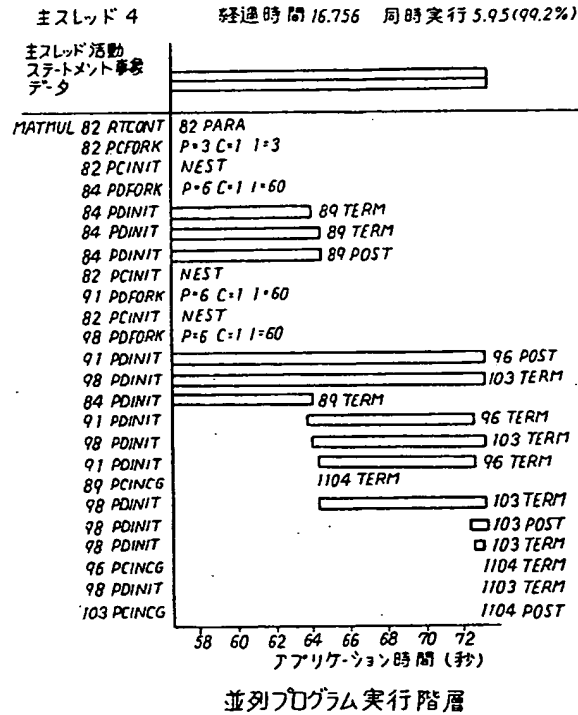
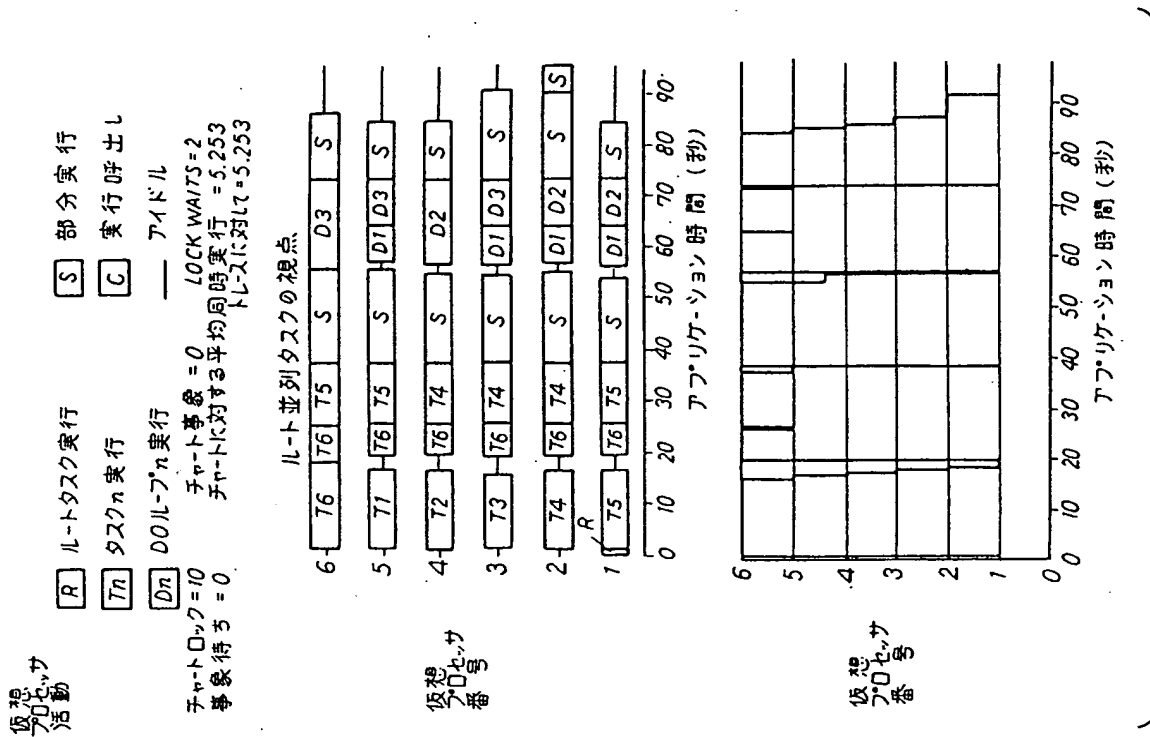
FIG. 5C

トレス解析表発生

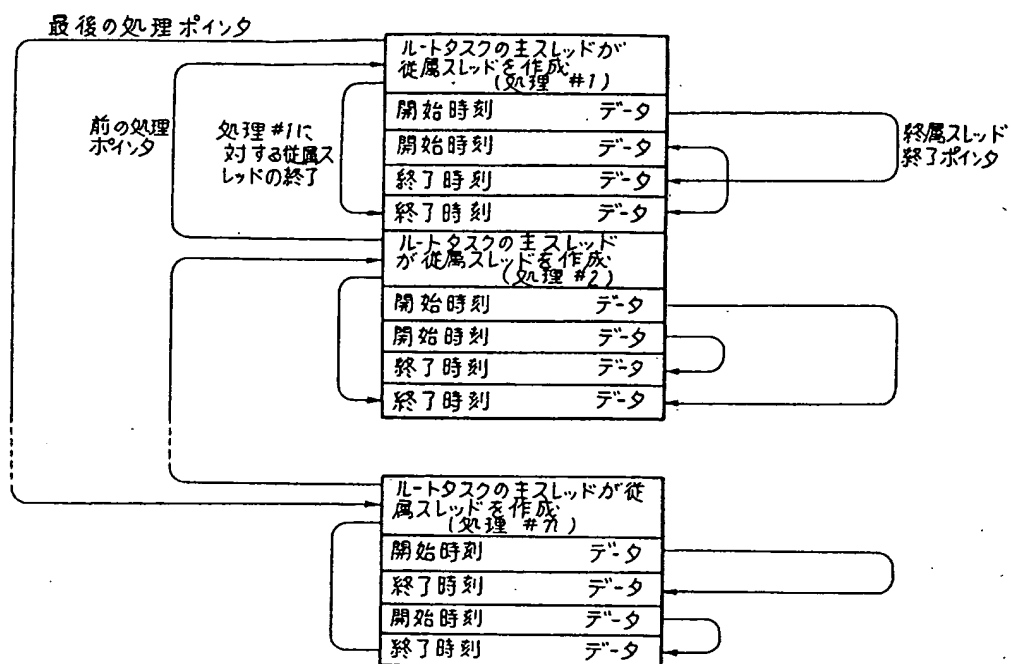
ビジュアル発生



特開平3-127236(16)



特開平3-127236(17)



並列活動表の要約

FIG. 10